Was ist agile Softwareentwicklung?

Nicht erst seit heute ist agile Softwareentwicklung ein wichtiges Thema. In den letzten Jahren wurden die klassischen Arten, wie Software geplant, diskutiert und umgesetzt wird, von neuen Methoden des Projektmanagements abgelöst. Die agile Softwareentwicklung ist eine davon. Aber was ist agile Softwareentwicklung? Das möchten wir im folgenden Blogbeitrag ein wenig durchleuchten.

Definition agile Softwareentwicklung

Agilität ist ein Merkmal, wie Organisationen und Projekte gehandhabt werden können. Es beinhaltet eine gewisse Flexibilität, Proaktivität und initiatives Handeln. Diese Merkmale wurden in den letzten Jahren auch Teil des Projektmanagements bei der Planung und Umsetzung von Softwareprojekten. Daraus entstand die agile Softwareentwicklung. Es gibt verschiedene Projektmanagementmethoden, die agile Ansätze beinhalten. Wir bei soom-it arbeiten oft mit einer abgewandelten Form von Scrum. insgesamt sollen die agilen Methoden zu einem schnelleren Einsatz der Software und zu erhöhter Flexibilität und Transparenz führen.

Früher war vor allem der Wasserfallansatz bei Softwareprojekten weit verbreitet. Hierbei werden zu Beginn komplexe Anforderungskataloge erstellt, die nicht so einfach zu ändern sind. Anschliessend wurden diese fixen Vorhaben in die Tat umgesetzt. Das hatte den Nachteil, dass auf verändernde Umweltbedingungen nicht so gut eingegangen werden kann.

Möglicher Ablauf agile Softwareentwicklung

Der hier beschriebene Ablauf ist nicht unbedingt modellhaft und wird wohl so nicht in einem Lehrbuch vorkommen. Aber es ist die Art, wie wir bei soom-it versuchen, Projekte umzusetzen und wir haben gute Erfahrungen damit gemacht. Nicht immer führen wir alle Schritte durch. Das hängt sehr stark vom Projekt und den Kundenbedürfnissen ab. Immer definieren wir User Stories, priorisieren diese und setzen sie in Sprints um, die wir vorgängig mit den Kunden besprechen.

Ziele und Probleme definieren, die mit der Software gelöst werden sollten.
Was soll grundsätzlich mit der Software erreicht werden und wie kann man das in einem Satz ausdrücken? Welche Probleme werden für den Endnutzer gelöst? Das sind wichtige, grundlegende Fragen in der Softwareentwicklung und es lohnt sich, sich immer wieder auf diese Fragen zurück zu besinnen.

Nutzer und Anspruchsgruppen identifizieren
Wer ist der Endnutzer der Software? Gibt es vielleicht unterschiedliche Gruppen, die die Software nutzen sollen? Welche Anspruchsgruppen sind sonst noch wichtig, damit das Projekt ein Erfolg wird?

Personas kreieren
Personas sind modellhafte Darstellungen von Nutzern. Sie werden z.B. auch im Marketing verwendet, um die Zielgruppe eines Produkts visibiler zu machen. Dazu werden Eigenschaften, Bedürfnisse und Nutzungsverhalten beschrieben. Personas eignen sich wunderbar als Diskussionsgrundlage für Softwareprojekte. Sie helfen auch Lösungen zu finden, die stark auf die Endnutzer ausgerichtet sind.

Bedürfnisse der Nutzer herausfinden
Um die Bedürfnisse der Nutzer herauszufinden, können Interviews geführt oder Nutzer bei der Handhabung einer Software beobachtet werden.

Risiken und Erwartungen besprechen
Eine gute Methode, um Risiken einen Softwareprojekts zu erkennen, ist die Premortem-Übung. Sie definiert den schlimmstmöglichen Zustand, der eintreten kann. Daraus werden dann die Schritte abgeleitet, wie es dazu kommen konnte. So kann man Erkenntnisse gewinnen, was jetzt getan werden kann, um zukünftige Schwierigkeiten zu verhindern.

MVP (Minimum Viable Product) definieren
Welches ist das kleinstmögliche Produkt, dass ich umsetzen möchte und anschliessend testen kann? Ein MVP bei einer App enthält z.B. nur die wichtigsten Kernfunktionen. Statt eine voluminöse Software zu bauen, wird nur ein kleiner Teil in einem ersten Schritt umgesetzt. Ein MVP ist kein Prototyp, sondern eine echte Lösung, die für sich alleine funktioniert. Diese kann anschliessend an der Zielgruppe getestet und Schritt für Schritt

verbessert werden. So kann schneller und direkt am Produkt gelernt und Verbesserungen initialisiert werden. Das senkt die Kosten und verbessert das Endprodukt. Die Kunst dabei ist es, Dinge wegzulassen, die Software also nicht unnötig auf zu blasen.

User Stories erstellen

User Stories sind "Anwendungsgeschichten" aus Nutzersicht, die beschreiben, was ein Nutzer konkret macht, wenn er eine Software nutzt. Software-Anforderungen werden also in eine Alltagssprache umformuliert. Dazu gibt es eine typische Form:

Als <Nutzergruppe>, möchte ich <gewünschte Anforderung> damit ich <Ziel/Nutzen>.

Beispiel:
Als <Wahlberechtigter in der Schweiz> möchte ich <wissen, welcher Politiker zu meinen politischen Präferenzen passt> damit <ich eine bessere Grundlage für meinen Wahlentscheid habe.>

Es gibt noch verschiedene Abgrenzungen (Epics und Themes) bei der Gestaltung und Akzeptanzkriterien für die Umsetzung der User Stories. Das soll aber nicht Teil dieses Blogbeitrags sein. Wer mehr darüber lesen will, findet hier gute Infos.

Nutzerbedürfnisse priorisieren

Das Product Backlog ist ein wichtiges Element agiler Softwareprojekte: Es enthält alle Arbeiten, die für die Erstellung einer erfolgreichen Anwendung gebraucht werden. Meist besteht das Ziel des Projekts auch darin, möglichst schnell ein erstes nutzbares Softwareprodukt zu haben. Damit dies gelingt, müssen die Arbeiten priorisiert werden. Das passiert entweder in Diskussion mit dem Kunden (Product Owner) oder kann mit spezifischen Modellen zur Priorisierung passieren (z.B. das Kano-Modell).

In Sprints die Software umsetzen

Das Konzept der Sprints ist Teil der agilen Entwicklungsmethode Scrum. In einer gewissen Zeit (2-4 Wochen) werden die vorher priorisierten und definierten User Stories umgesetzt. Nach dieser Zeit treffen sich Kunden, Entwickler und Scrum Master, um die Ergebnisse zu diskutieren und die Anforderungen zu definieren, die im nächsten Sprint

umgesetzt werden sollen. Das führt zu mehr Transparenz und Mitsprache für den Kunden und eine agilere Umsetzung des Projekts.

Vorteile agiler Softwareentwicklung

- Bei agilen Projekten wird nicht im Vornherein alles bis ins kleinste Detail definiert. Das führt zu mehr Flexibilität und das Projekt kann relativ schnell starten.

- Schnellerer Einsatz der Software: Da nur die wichtigsten Anforderungen definiert werden und auch "on-the-go" angepasst werden können, wird die Software schneller zum Einsatz kommen. Es muss nicht alles bis zum bitteren Ende durchgeboxt werden. Laufend werden Anforderungen getestet und abgenommen.

- Die Nutzer werden, wenn möglich, mit einbezogen. So wird von Beginn weg Software erstellt, die dem Nutzer dient und nicht nur dem Entwickler oder Besitzer.

- Kostenvorteile: Oft entstehen gesamthaft tiefere Kosten, da Fehler weniger schlimm sind, als wenn alles in einem Guss durchgezogen wird wie bei der Wasserfallmethode. Anpassungen können schneller gemacht werden und das verringert die Kosten.

- Transparenz für Kunden und Entwickler: Resultate sind schnell sichtbar und können getestet und Änderungswünsche eingebracht werden.

Nachteile agiler Softwareentwicklung

- Kosten nicht so gut planbar. Da sich Anforderungen ändern können, sind die Kosten nicht immer so klar voraussehbar. Häufig wird mit einem Kostendach oder nach Aufwand gearbeitet. Das ist aber nicht für alle Kunden interessant, weil sie oft gerne einen fixen Preis für eine fixe Leistung hätten.

- Hohe Anforderungen an Kommunikation auf beiden Seiten. Agile Methode haben einen erhöhten Kommunikationsaufwand, da jede User Story einzeln angeschaut

und bestätigt wird. Die Beteiligten am Projekt treffen sich regelmässig und tauschen sich aus, priorisieren und entscheiden.

- Erhöhter Testing-Aufwand. Bei agilen Methoden wird immer wieder getestet. Das führt zu einem höheren Aufwand in diesem Bereich.

Unsere Erfahrungen

Wir nutzen agile Methoden seit vielen Jahren und es ist unsere präferierte Art zu arbeiten und Softwareprojekte umzusetzen. Wir setzen dabei die Entwicklungsmethode Scrum sehr pragmatisch um ohne uns an alle formalen Vorgaben zu halten. Uns gefällt die aktive Diskussion mit den Kunden und die erhöhte Transparenz unserer Arbeit. Wir können Kunden bei der Priorisierung beraten und unsere Einschätzung zum Aufwand der Umsetzung einer User Story aus technischer Sicht abgeben. Zusammen mit der eher funktionalen Kundensicht ergeben sich so gute Lösungen, die den Programmieraufwand verringern können.

Weiter finden wir, dass Software vor allem für die Endnutzer entwickelt wird und nicht nur den Präferenzen der Auftraggeber und Entwickler entsprechen sollten. Agile Projektmanagementmethoden versuchen genau das, indem sie die Bedürfnisse der Zielgruppe stark mit einbeziehen. Es gibt zudem den Entwicklern die Freiheit, die Software-Anforderungen auf ihre eigene Art zu entwickeln und nicht zu stark an Vorgaben gebunden zu sein. Agile Entwicklung funktioniert zudem gut, wenn die Kunden einfach mit einer Idee zu uns kommen und noch nicht alles bis ins Detail definiert haben. So kann schnell gestartet und auch innert nützlicher Frist erste, testbare Lösungen präsentiert werden.

Aus unserer Erfahrung sind die grössten Herausforderungen, sich Zeit zu nehmen, gute User Stories zu schreiben, richtig zu priorisieren und ein gemeinsames Verständnis für agiles Vorgehen mit den Kunden zu finden. Agile Softwareentwicklung fordert von allen Projektbeteiligten ein hohes Engagement für das Projekt und eine gemeinsame Kommunikationsbasis.